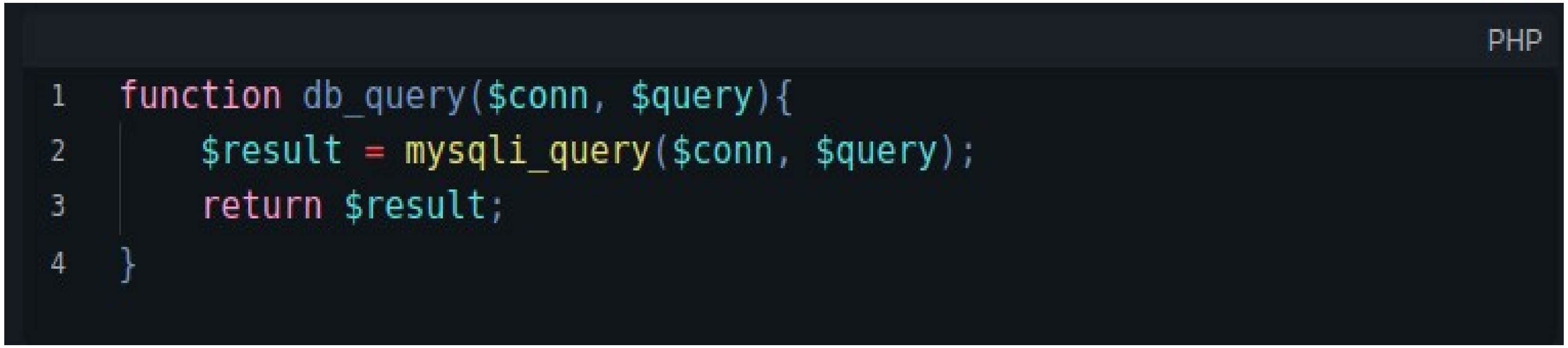
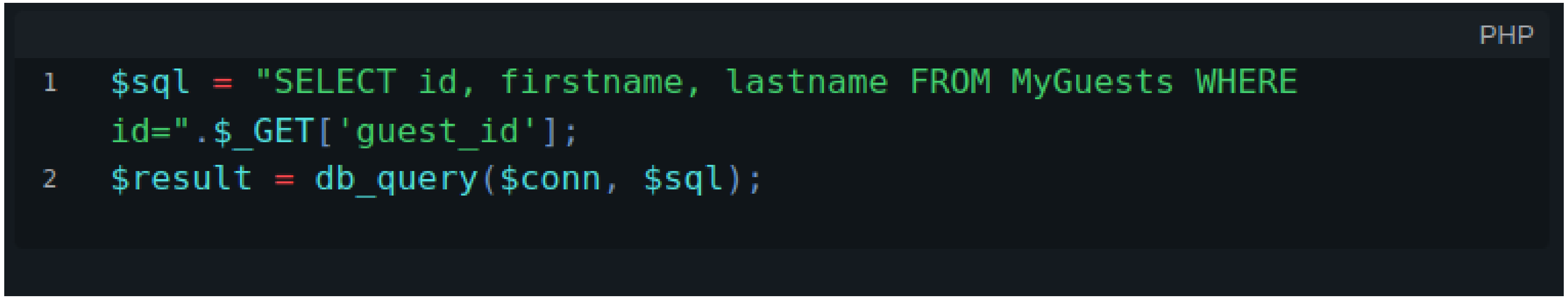
Secure Coding Review

Manual Code Review:



```php
function db_query($conn, $query){
    $result = mysqli_query($conn, $query);
    return $result;
}
```

Here we can see that `mysqli_query()` is wrapped into the `db_query()` function, and that the `$query` parameter is passed directly without modification.
It is very common for functions to be nested into other functions, so simply analysing the local context of a function is sometimes not enough to determine if a vulnerability is present. We now need to trace the uses of the `db_query()` function throughout our code to identify potential vulnerabilities.

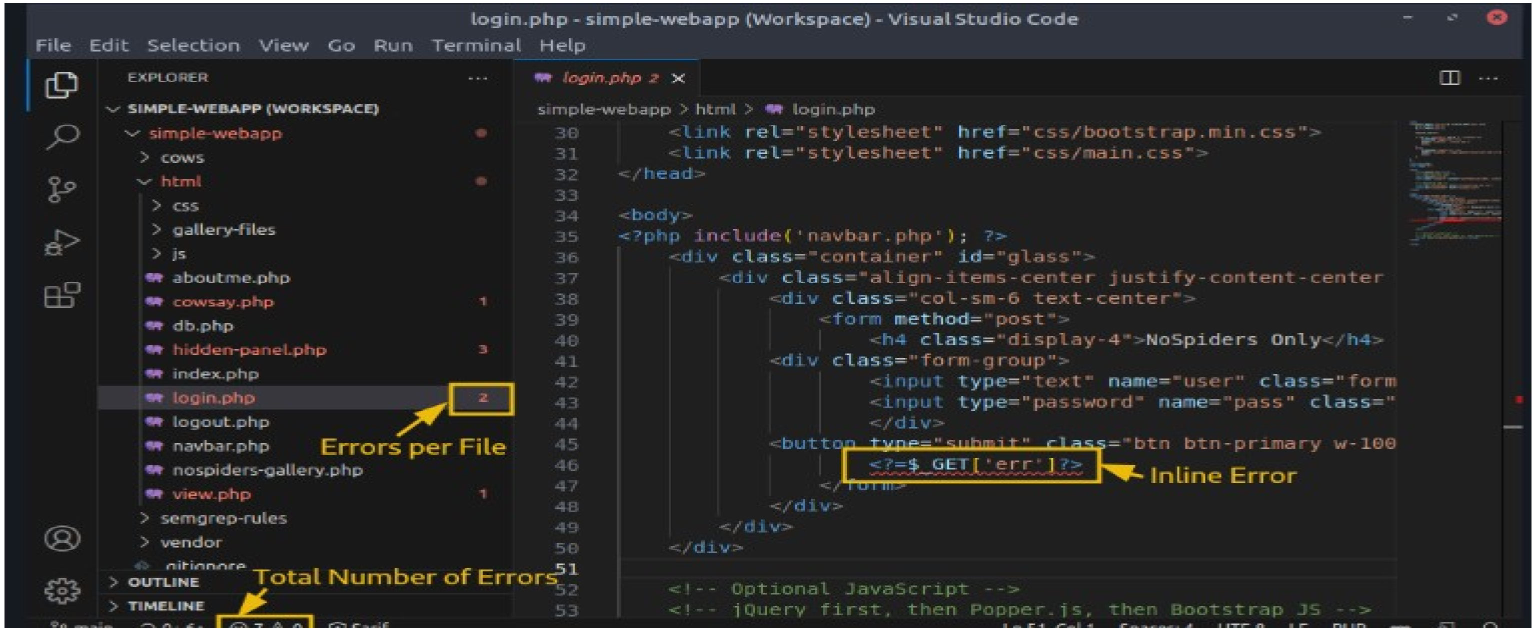


```php
$sql = "SELECT id, firstname, lastname FROM MyGuests WHERE id=".$_GET['guest_id'];
$result = db_query($conn, $sql);
```

Here’s a SQL injection! Whatever is passed in the `guest_id` parameter via the GET method will be concatenated to a raw SQL query without any input sanitisation, enabling the attacker to change the query.

You Can Use A Plugin To Secure Your Code:

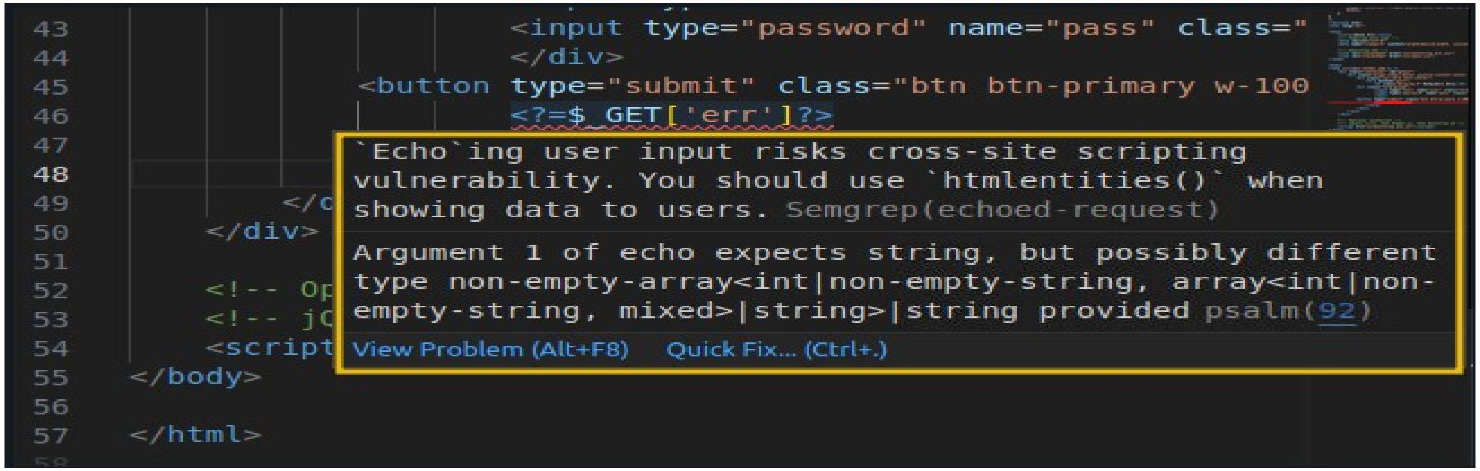
Psalm: Is a tool supports IDE, will check anything you type in real-time and show you the alerts



```
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <link rel="stylesheet" href="css/main.css">
</head>

<body>
<?php include('navbar.php'); ?>
    <div class="container" id="glass">
        <div class="align-items-center justify-content-center
            <div class="col-sm-6 text-center">
                <form method="post">
                    <h4 class="display-4">NoSpiders Only</h4>
            <div class="form-group">
                    <input type="text" name="user" class="form
                    <input type="password" name="pass" class="
                    </div>
            <button type="submit" class="btn btn-primary w-100
                    <?=$_GET['err']?>
                </form>
            </div>
        </div>
    </div>

    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
```



```
                    <input type="password" name="pass" class="
                    </div>
            <button type="submit" class="btn btn-primary w-100
                    <?=$_GET['err']?>

            </c
        </div>

    <!-- Op
    <!-- jQ
    <script
</body>

</html>
```

`Echo`ing user input risks cross-site scripting vulnerability. You should use `htmlentities()` when showing data to users. Semgrep(echoed-request)

Argument 1 of echo expects string, but possibly different type non-empty-array<int|non-empty-string, array<int|non-empty-string, mixed>|string>|string provided psalm(92)

Thank You